DE4106793

Publication Title:

Communications interface for open communications between users and layered system - is based on macro library and contains four groups of procedures for session, link, data processing and user coded procedures

Abstract:

Abstract of DE 4106793

(A1) Translate this text A communication interface for open communications between users and components of a distributed communications system is based on a macro library and defined by four different groups of procedures. The first group contains session initialising and termination and channel opening and closing procedures accessible by a subscriber. The second group contains procedures accessible by an active partner to set up and terminate a link to a passive partner and to initiate data exchange. The third group is accessed by a passive partner to service incoming data and respond to remotely initated exchanges. The fourth group contains user coded procedures called up from the macro library, e.g. specifiying a main menu. USE/ADVANTAGE - Communications system with layered structure. The open, convenient interface has high application related functionality and small number of procedures with uniform access to different protocols.

---

(19) BUNDESREPUBLIK DEUTSCHLAND

DEUTSCHES PATENTAMT

(12) **Offenlegungsschrift**

(10) **DE 41 06 793 A1**

(51) Int. Cl.[5]:
G 06 F 13/14
H 04 L 29/06

(21) Aktenzeichen: P 41 06 793.2
(22) Anmeldetag: 4. 3. 91
(43) Offenlegungstag: 17. 9. 92

DE 41 06 793 A1

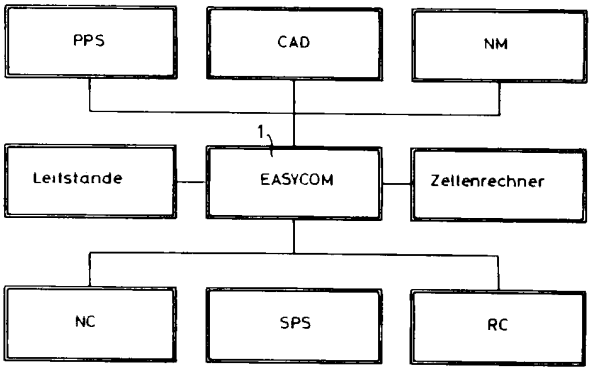
(71) Anmelder:
Licentia Patent-Verwaltungs-GmbH, 6000 Frankfurt, DE

(72) Erfinder:
Baudisch, Matthias, Dipl.-Ing., 7332 Eislingen, DE

Prüfungsantrag gem. § 44 PatG ist gestellt

(54) Kommunikationsschnittstelle

(57) Gegenstand der Erfindung ist eine Kommunikationsschnittstelle für die offene Kommunikation von Anwendern mit Komponenten eines verteilten Kommunikationssystems, das in Schichten strukturiert ist. Die Kommunikationsschnittstelle basiert auf einer MACRO-LIBRARY und enthält verschiedene Gruppen von Prozeduren wie Initialisierung oder Beendigung einer Kommunikation, Öffnung oder Schließung eines logischen Kommunikationskanals, Aufbau und Abbau einer Kommunikationsverbindung und Bedienung ankommender Informationen.



Berücksichtigte Anwendungsbereiche von EASYCOM

DE 41 06 793 A1

Beschreibung

Die Erfindung bezieht sich auf eine Kommunikationsschnittstelle für die offene Kommunikation von Anwendern mit Komponenten eines verteilten Kommunikationssystems, das in Schichten strukturiert ist. Für die Realisierung des offenen Verbunds von Rechnerkomponenten unterschiedlicher Herstellung wurde als Basis das ISO-/OSI-Referenzmodell genormt (ISO 7498), das Kommunikationssysteme durch Unterteilung in Schichten strukturiert. Das ISO-/OSI-Referenzmodell setzt sich aus sieben Schichten zusammen, für die Standards definiert sind. Jeder Schicht sind Aufgaben von Kommunikationsfunktionen zugeordnet. Eine Reihe von Implementierungen der Standards sind kommerziell verfügbar. Die siebte Schicht umfaßt die anwendungsspezifischen Dienste für Kommunikationsanwendungen.

Für die offene Kommunikation im industriellen Bereich wurde aus den Normen für jede Schicht das geeignete Protokoll und die geeigneten Dienste ausgewählt und zu einem Kommunikationssystem zusammengefaßt. Ein wichtiges Protokoll unter diesen Kommunikationssystemen ist MAP (Manufacturing Automation Protokoll), das für die Schichten des ISO-/OSI-Referenzmodells die Normen oder Untermengen von Normen angibt.

Die siebte Schicht der MAP-Kommunikationsarchitektur ist in sich nochmals unterteilt. Eine obere Schicht enthält Applikationsprotokolle, z. B. File Transfer, Access und Management zum Filetransfer, Manufacturing Message Specification (MMS, ISO 9506) zur Kommunikation in der industriellen Fertigung sowie Directory Services, Remote Operation Service Element und Network Management. Diesen Protokollen ist eine Schicht mit ACSE (Application Control Service Element) Diensten für den Verbindungsaufbau (A-Associate), Verbindungsabbau (A-Release) und Verbindungsabbruch (A-Abort) unterlagert.

Die Akzeptanz von international genormten Kommunikationsprotokollen, wie MAP oder TOP, ist bei Anwendern vielfach noch gering. Die Anbieter von Produkten für die Fertigungsautomatisierung sind daher gezwungen, eine Vielzahl von Kommunikationsprotokollen bzw. -profilen, wie MMS, TCP, IP, SINEC, Decnet, SNA, zu unterstützen.

Weiterhin sind die Applikationsschnittstellen häufig komplex. Die MMS-Applikationsschnittstelle weist z. B. 240 Library-Prozeduren und 170 vordefinierte, vom Anwender zu schreibende Prozeduren auf. Diese sind vom jeweiligen Anwender nur mit einem hohen Aufwand an Spezialisten beherrschbar. Es ist deshalb die Aufgabe der Erfindung, eine komfortable, offene Schnittstelle mit hoher anwendungsnaher Funktionalität und geringer Anzahl von Prozeduren zu entwickeln, die für Application (z. B. Leitstände, Zellenrechner, Benutzeroberflächen) einen einheitlichen Zugang zu unterschiedlichen Kommunikationsprotokollen ermöglicht.

Die Aufgabe wird erfindungsgemäß dadurch gelöst, daß die auf einer Macro-Library basierende Kommunikationsschnittstelle in Verbindung mit der obersten Schicht durch vier verschiedene Gruppen von Prozeduren bestimmt ist, von denen die erste Gruppe die von einem passiven oder aktiven Kommunikationsteilnehmer aufrufbare Prozeduren zur Initialisierung oder Beendigung einer Kommunikation sowie zur Öffnung oder Schließung eines logischen Kommunikationskanals enthält, wobei die zweite Gruppe nur von einem aktiven Kommunikationspartner aufrufbare Prozeduren mindestens zum Aufbau und Abbau einer Kommunikationsverbindung zu einem passiven Kommunikationspartner und zur Initiierung eines Datenaustausches enthält und wobei die dritte Gruppe nur von einem passiven Kommunikationsteilnehmer aufrufbare Prozeduren zur Bedienung ankommender Informationen und Beantwortung eines entfernt initiierten Datenaustausches aufweist, während die vierte Gruppe von einem Anwender zu kodierende Prozeduren, die von der Macro Library aufgerufen werden, aufweist und die Prozeduren zur Spezifizierung eines Hauptmenüs, einer Reaktion auf eine bestimmte Eingabe und zur Angabe auszuführender Funktionen enthält. Diese Macro-Schnittstelle geht von der Programmierzugangsschnittstelle der obersten vorhandenen Schicht des jeweiligen Kommunikationsprofils aus. Beispielsweise besitzt das Kommunikationsprofil 7-layer Full Map 3.0 als oberste Schicht das Applikationsprotokoll MMS mit dem Programmierinterface MMS EASE, auf das die Macro-Kommunikationsschnittstelle aufsetzt.

Durch die vorstehend beschriebene Macro-Schnittstelle werden Standardaufrufe zur Verfügung gestellt, die sich auf die Initialisierung bzw. das Beenden der Kommunikation, die An- bzw. Abmeldung eines Kommunikationswunschs, den Auf- bzw. Abbau einer Kommunikationsverbindung und die Abwicklung des Datenverkehrs über besondere Aufrufe beziehen.

Letztere bieten Funktionalität mit Macro-Charakter, aufgeschlüsselt in Standardgeräteoperationen. Die Macro-Kommunikationsschnittstelle erfüllt das Prinzip der Offenheit, d. h. sie ist herstellerunabhängig bezüglich der verwendeten Kommunikationsprodukte (Controller Boards, Treiber, Protokoll-Software) und unterstützt offene Kommunikationsprotokolle bzw. -profile gemäß der OSI-Norm. Zusätzlich können auch nichtoffene Kommunikationsprotokolle bzw. -profile (Transmission Control Protocol, Digital Network Architecture) über eine einheitliche Schnittstelle zugänglich gemacht werden. Die Kommunikationsschnittstelle ist darüber hinaus offen in dem Sinne, als sie Kommunikation über ein bestimmtes Protokoll auch mit solchen Partnerstationen erlaubt, die nicht mit dieser Schnittstelle ausgerüstet sind. Die MACRO-Schnittstelle erlaubt die Erstellung einer von dem unterlagerten Kommunikationsprotokoll bzw. -profil unabhängigen Applikation, so daß diese beim Austausch des verwendeten Kommunikationsprotokolls unverändert bleiben kann. Damit ist eine Migration, d. h. Übergang von firmenspezifischen zu offenen Kommunikationsprofilen durch einen Kunden möglich. Ferner ist die mit der MACRO-Schnittstelle erstellte Applikation unabhängig von der eingesetzten Netzwerktopologie, die z. B. sternförmig, busförmig, ringförmig oder Punkt-zu-Punkt-Verbindung sein kann. Die MACRO-Schnittstelle gibt dem jeweiligen Kommunikationsteilnehmer keine Angaben darüber, ob sich sein Kommunikationspartner auf derselben (maschineninterne Kommunikation) oder auf einer anderen Maschine (Kommunikation übers Netz) befindet, da diese Kommunikationsbeziehungen der Projektierung unterliegen. Die Abhängigkeit vom jeweiligen Betriebssystem (DOS, UNIX/REALIX, RMX, OS2, VMS) ist gering.

Es ist vorgesehen, daß drei Prozeduren in Form von LIBRARY-CALLS vorgesehen sind, von denen zwei auf

Standarddatentypen und Standardgeräteoperationen und die andere auf Datenbankoperationen bezogen ist. Die Prozeduren für Standardgeräteoperationen sind insbesondere auf Operationen mit DIRECT-NUMERICAL-CONTROL-Funktionalität und Betriebs- bzw. Maschinendatenerfassung gerichtet. Es kann sich zweckmäßigerweise um Datenaustausch folgender Art handeln:

- Übertragen/Rückübertragen von Auftragsdaten, NC-Programmen, Werkzeugkorrekturdaten, RC-Programmen usw.
- Starten von NC-Programmen, RC-Programmen usw.
- SPS-Eingänge/Ausgänge/Merker setzen bzw. lesen
- Spindeldrehzahl, Vorschub auslesen
- Maschinenzustände, Störungen melden
- Stückzahlen/Ausschuß melden.

In einer bevorzugten Ausführungsform sind außer diesen Grunddatenaustauscharten auch sehr komplexe wie Event und Alarmhandling vorgesehen.

Die Erfindung wird im folgenden anhand eines in einer Zeichnung dargestellten Ausführungsbeispiels näher beschrieben, aus dem sich weitere Einzelheiten, Merkmale und Vorteile ergeben. Es zeigt

Fig. 1 ein Schema der Beziehungen zwischen einer Kommunikationsschnittstelle für die offene Kommunikation mit Applikationen und unterlagerten Kommunikationsprotokollen und -profilen;

Fig. 2 ein Übersichtsschaltbild über die Anwendungsgebiete, die die Kommunikationsschnittstelle gem. **Fig.** 1 unterstützt;

Fig. 3 Details der unterlagerten Kommunikationsprofile gem. **Fig.** 1.

Eine einheitliche Kommunikationsschnittstelle **1** mit hoher anwendungsnaher Funktionalität und geringer Anzahl von Prozeduren steht für eine Applikation **2** zur Verfügung, um dieser den Zugang zu Kommunikationsprotokollen bzw. -profilen **3, 4, 5, 6, 7, 8, 9, 10** zu ermöglichen. Bei den Kommunikationsprotokollen bzw. -profilen **3** bis **6** kann es sich um offene oder nichtoffene Protokolle oder Profile handeln. In **Fig.** 1 sind jeweils die Kommunikationsprotokolle bzw. -profile für MMS, die MAP-Applikationsdienste für die industrielle Fertigung, für SINEC, TCP/IP, X.25, SNA, DECnet, LAT und XNS angegeben. MMS ergibt sich z. B. aus ISO/DIS 9506-1, 9506-2 Manufacturing Message Specification. SINEC ist im SINEC AP 1.0 Handbuch beschrieben. TCP, Transmission Control Protocol ergibt sich aus RFC 793, 813, 879, 814, 816, 817, 889, 896 und 964 (RFC = Standards for Internet). XNS ist ein Protokoll der Firma XEROX. DEc net ist die Digital Network Architekture z. B. Phase IV).

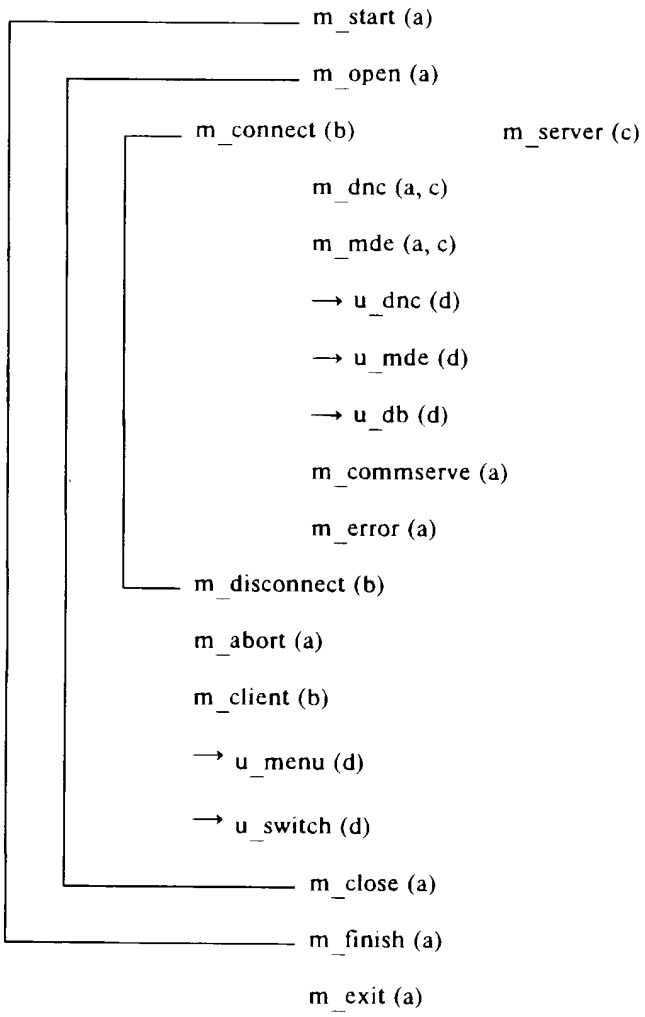
Die Kommunikationsschnittstelle **1** enthält Macro-Libraries **11, 12, 13, 14**, die jeweils mit Libraries **15, 16, 17, 18** zusammenwirken, die den Kommunikationsprotokollen bzw. -profilen **3** bis **10** zugeordnet sind.

Mit der Kommunikationsschnittstelle **1** können darüber hinaus die in **Fig.** 2 gezeigten Anwendungen unterstützt werden. Es handelt sich um die Kopplungen zwischen den CIM-Komponenten PPS, CAD, zu Leitständen, Zellenrechnern und zu NC-Systemen, RC-Systemen und SPS-Systemen, bzw. Kopplung zwischen Netzwerkmanagementsystemen und den einzelnen Netzstationen zur Netzüberwachung.

Die **Fig.** 3 zeigt Protokollarchitekturen und Kommunikationsprofile **19** für Mini MAP, **20** für MAP, **21** für Top, **22** für TCT/IP, **23** für SINEC, **24** für SNA und **25** für DEC net.

Die über ein Kommunikationsnetz mittels der Kommunikationsschnittstelle **1** angesprochenen Objekte müssen zunächst projektiert werden. Die Projektierungsinformation wird in Konfigurationsfiles abgelegt. Dazu können in einer bevorzugten Ausführungsform Projektierungstools mit menügesteuerter Benutzerführung verwendet werden.

Die MACRO-Kommunikationsschnittstelle **1** umfaßt in der Sprache C folgende Prozeduren:



Diese Prozeduren sind im folgenden ausführlich beschrieben und gliedern sich in zwei Hauptgruppen:

Kommunikationsverwaltungsdienste

Hierbei handelt es sich um Standardaufrufe zum Initialisieren bwz. Beenden der Kommunikation (m_start, m_finish, m_exit), An- bzw. Abmelden eines Kommunikationswunsches (m_open, m_close), Auf- und Abbau einer Kommunikationsverbindung (m_connect, m_disconnect, m_abort) und zur Behandlung von Kommunikationsfehlerfällen (m_error).

Anwendungsdienste

Über sie erfolgt die Abwicklung des eigentlichen Datenverkehrs. Dies geschieht sehr applikationsnah. Funktional wird unterschieden nach DNC-Operationen (m_dnc, u_dnc), MDE/BDE-Operationen (m_mde, u_mde), Datenbank-Operationen (u_db) und Operationen, die die Bedienung und Visualisierung betreffen (u_menu, u_switch). Die DNC bzw. MDE/BDE Operationen gliedern sich in sogenannte Standardgeräteoperationen (z. B. DOWNLOAD, UPLOAD, READ, WRITE usw.), die auf Standarddatentypen (NC-PROGRAMM, MERKER usw.) angewendet werden. Zu den Datenbankoperationen gehören z. B. Abfrage nach Tabelleneinträgen, Hinzufügen neuer Einträge (ASK_DNC, ADD_MDE usw.). Für das Bedienen von ankommenden Kommunikationswünschen stehen je nach gewünschter Rolle entsprechende Dienste zur Verfügung (m_comm_serve, m_client, m_server).

Die Prozeduren lassen sich in vier Gruppen einteilen:

(a) Funktionen, die sowohl vom aktiven als auch vom passiven Kommunikationspartner aufgerufen werden, um einen Dienst zu nutzen;
(b) Funktionen, die nur vom aktiven Kommunikationspartner (Client, Requester) aufgerufen werden, um einen Dienst zu nutzen;
(c) Funktionen, die nur vom passiven Kommunikationspartner (Server, Responder) aufgerufen werden, um

einen Dienst zu nutzen;
(d) Funktionen, die von der Kommunikationsschnittstelle 1 für den Anwender aufgerufen werden, die aber vom Anwender codiert werden müssen.

Die Kommunikationsverwaltungsdienste beinhalten die nachstehend beschriebenen Prozeduren:

m-start

Mit m_start wird eine Applikation bezüglich der Kommunikation initialisiert. Dazu gehört das Einlesen der Projektierinformation aus den Konfigurationsdateien, das Setzen von Kommunikationsparametern sowie das Bekanntmachen der anzusprechenden Geräte und deren zugreifbare Objekte (z. B.: die SPS mit der Busadresse 7711 besitzt die und die Merker, Zähler, Ein- und Ausgänge, die NC mit der Busadresse 4713 besitzt die und die Register für Spindeldrehzahl, Vorschubgeschwindigkeit usw.). Diese Funktion muß vor allen anderen Diensten sowohl vom aktiven als auch vom passiven Kommunikationspartner aufgerufen werden.

m-finish

Die Funktion m_finish stellt das Gegenstück zu m_start dar. Sie dient zur Beendigung der Kommunikation. Sie sorgt dafür, daß die Kommunikationshard- und -software in einen definierten Zustand zurückgesetzt wird. Dazu gehört der Abbruch aller bestehenden Verbindungen, das Abmelden aller angemeldeten Kommunikationskanäle, die Rückgabe von belegtem Speicherplatz usw. Diese Funktion muß sowohl vom aktiven als auch vom passiven Kommunikationspartner vor dem Verlassen des laufenden Programms aufgerufen werden. Nach Aufruf dieser Funktion muß, bevor andere Dienste wieder zur Verfügung stehen, erneut mit start aufgerufen werden. Wird ein Applikationsprogramm ohne durchlaufen von m_finish verlassen, kann es passieren, daß der betreffende Netzknoten erst nach einem Knotenneustart wieder ansprechbar ist.

m-exit

Mit m_exit kann ein Applikationsprogramm jederzeit definiert verlassen werden. Sie beinhaltet die oben beschriebene Funktion m_finish. Sie kann z. B. direkt als Signal Handler Routine für das Tastatursignal Control_C verwendet werden.

m-open

Mit m_open wird ein logischer Kommunikationskanal geöffnet, d. h. ein lokaler Applikationsprozeß meldet sich mit seinem logischen Namen (local application name) beim Kommunikationssystem an. Dieses reserviert dem Applikationsprozeß die für die Kommunikation notwendigen Ressourcen und vergibt eine logische Kanalnummer (channel). Die Anmeldung mittels m_open ist eine reine lokale Angelegenheit, welche die Voraussetzung für die weitere Kommunikation (Verbindungsaufbau, Datenaustausch) mit entfernten Partnern unter der zugewiesenen Kanalnummer schafft.
Die Applikation kann verschiedene Typen von Kommunikationskanälen öffnen:
über den Kanal soll aktiv eine Verbindung zu einer entfernten Partnerapplikation aufgebaut werden (Connect Channel), über den Kanal sollen Broadcoast- oder Multicast-Meldungen abgesetzt werden (Broadcast Channel, Multicast Channel), an dem Kanal soll auf eingehende Kommunikationswünsche von entfernten Partnern gewartet werden (Listen Channel). Jeder dieser Typen kann ggf. noch als beschleunigter Kanal auftreten (Expedited Connect Channel, Expedited Broadcast Channel, Expedited Multicast Channel, Expedited Listen Channel).
Ein Beispiel dafür wäre MAP 3.0, das die Möglichkeit zuläßt, zur Erhöhung der Performance statt über 7 Schichten nur über 3 Schichten (Mini MAP) zu kommunizieren. Die Programmierung von m_open muß in Einklang mit der Projektierung stehen, d. h. beispielsweise, daß für den application name in einer Adreßtabelle ein gültiger Eintrag vorhanden sein muß. Diese Funktion muß sowohl vom aktiven als auch vom passiven Kommunikationspartner vor jeglicher weiterer Kommunikation aufgerufen werden. Es können mehrere Kanäle jeden Typs parallel geöffnet sein. Solange die Zahl der gewünschten Verbindungsmöglichkeiten geringer als die maximale Anzahl der vergebbaren Kanäle ist, kann man die Kommunikationskanäle statisch verwenden, d. h. einmalig öffnen und dann geöffnet lassen. Es gibt aber auch die Möglichkeit, die Kanäle dynamisch zu verwenden, d. h. vor dem Kommunikationswunsch öffnen und danach wieder schließen.

m-close

Die Funktion m_close stellt das Gegenstück zu m_open dar. Sie dient zum Schließen eines logischen Kommunikationskanals, bzw. Abmelden eines lokalen Applikationsprozesses. Sie sorgt für die Rückgabe der für die Kommunikation reservierten Ressourcen und kennzeichnet den betreffenden Kanal (channel) als frei. Die Abmeldung mittels m_close ist eine reine lokale Angelegenheit, welche die Voraussetzung für die Kommunikation (Verbindungsaufbau, Datenaustausch) mit entfernten Partnern unter der zugewiesenen Kanalnummer entzieht. Bei Auftreten eines erneuten Kommunikationswunsches muß erst wieder ein Kanal geöffnet werden. Diese Funktion steht sowohl dem aktiven als auch dem passiven Kommunikationspartner zur Verfügung.

m-connect

Mit m_connect kann eine Kommunikationsverbindung zu einer entfernten Partnerapplikation aufgebaut werden. Hierzu ist der zu verwendende Kanal und der Name der entfernten Station (remote application name) anzugeben. Vorausgesetzt wird, daß der aktive Kommunikationsparameter vorher einen Connect Channel, der passive einen Listen Channel geöffnet hat. Zum Absetzen von Broadcast-Meldungen ist kein Verbindungsaufbau notwendig. Während des Verbindungsaufbaus werden die Kommunikationsparameter, nach denen die Kommunikation erfolgen soll, ausgehandelt. Anschließend kann der eigentliche Datenaustausch erfolgen.

Die Programmierung von m_connect muß in Einklang mit der Projektierung stehen, d. h. beispielsweise, daß für den application name des Partners in einer Adreßtabelle ein gültiger Eintrag vorhanden sein muß. Für die Funktion m_connect ist es unerheblich, ob die angesprochene Partnerapplikation sich auf der selben (maschineninterne Kommunikation) oder einer anderen Maschine (Kommunikation übers Netz) befindet, d. h. die Schnittstelle erlaubt Interprozeßkommunikation. Solche Kommunikationsbeziehungen unterliegen der Projektierung. Der Applikationsprogrammierer hat die Möglichkeit, die Kommunikationsverbindungen statisch zu verwenden, d. h. einmalig aufbauen und dann aufgebaut lassen. Er kann aber die Verbindungen auch dynamisch verwenden, d. h. vor dem Datenaustausch aufbauen und danach wieder abbauen.

m-disconnect

Die Funktion m_disconnect stellt das Gegenstück zu m_connect dar. Der aktive Partner kann damit eine bestehende Kommunikationsverbindung mit einem passiven Partner gütlich abbauen. Voraussetzung aber ist, daß der Datenaustausch zwischen den beiden Applikationen komplett abgeschlossen ist, d. h. daß kein unbeantworteter Kommunikationswunsch mehr existiert. Die betreffenden Kanäle des aktiven Kommunikationspartners (Connect Channel) bzw. des passiven (Listen Channel) können dann für den Aufbau neuer Verbindungen genutzt werden.

m-abort

Die Funktion m_abort erlaubt sowohl dem aktiven als auch dem passiven Kommunikationspartner eine aufgebaute Verbindung jederzeit rigoros abzubrechen. Dabei werden aber Datenverluste in Kauf genommen. Deshalb sollte diese Funktion nur mit großer Sorgfalt, z. B. bei Auftreten grober Fehlerfälle angewendet werden. Die betreffenden Kanäle des aktiven Kommunikationspartners (Connect Channel) bzw. des passiven (Listen Channel) können dann für den Aufbau neuer Verbindungen genutzt werden.

m-error

Jede Prozedur der Kommunikationsschnittstelle **1** gibt bei Mißlingen einen Fehlercode zurück. Zu jedem dieser Fehlercodes kann man sich mittels m_error einen aussagefähigen Fehlertext auf die Standardfehlerkonsole ausgeben lassen.

Die Anwendungsdienste beinhalten die nachstehend angegebenen Prozeduren:
Die eigentliche Funktionalität (z. B. Download von NC-Programmen, Schreiben eines Merkers) wird von den Diensten m_dnc und m_mde zur Verfügung gestellt. Dazu ist vom aktiven Kommunikationspartner anzugeben, was (z. B. Download) er mit wem (z. B. Fertigungsleitrechner) kommunikativ austauschen will. Der passive Partner hat nur dafür zu sorgen, daß ankommende Kommunikationswünsche auch bedient werden. Dazu stehen die Funktionen m_comm_serve bzw. m_server sowie m_client zur Verfügung. Nach deren Aufruf wird die Applikation jeweils automatisch von der Schnittstelle informiert, wenn eine Dienstanforderung angekommen ist bzw. wenn ein Dienst erbracht worden ist (u_dnc, u_mde). Die daraufhin ausführende Aktion ist dann innerhalb von u_dnc bzw. u_mde vom Applikationsprogrammierer zu kodieren. Zusätzlich sind definierte Schnittstellen zu Datenbanken (u_db) bzw. zur Visualisierung und Bedienung (u_menu, u_switch) eingerichtet. Die Dienste m_dnc bzw. m_mde können vom aktiven Kommunikationspartner sowohl synchron als auch asynchron verwendet werden. Synchron heißt, die Applikation erhält die Kontrolle erst wieder von der Schnittstelle zurück, nachdem der Datenaustausch abgeschlossen ist, sie muß also bis nach der Datenübertragung warten, hat die Daten dann aber direkt zur Verfügung. Asynchron bedeutet, daß die Applikation nach dem Anstoß der Datenübertragung (z. B. UPLOAD_REQ) sofort wieder die Kontrolle von der Schnittstelle zurückerhält. Die Applikation läuft also weiter, d. h. sie kann während des Datenaustauschs andere Aufgaben ausführen. Sie ist allerdings für das Abholen der Daten selbst verantwortlich. Im asynchronen Fall kann der Benutzer bei Absetzen eines Requests (z. B. UPLOAD_REQ) eine Benutzerreferenz übergeben, die er mit der entsprechenden Confirmation (UPLOAD_CNF) unverändert zurück erhält. Auf diese Weise hat der Benutzer u. U. schneller Zugriff auf bestimmte Daten, weil es ihm einen Tabellensuchlauf ersparen kann. Der Inhalt der Referenz ist Benutzersache und kann beispielsweise eine Benutzererkennung, eine Unterprogrammiereinsprungadresse o. ä. sein. Im asynchronen wie im synchronen Fall kann der Benutzer eine Zeitdauer in sec mitgeben, die er maximal bereit ist auf die zu diesem Request gehörende Confirmation zu warten (TIMEOUT). In einer bevorzugten Ausführungsform ist vorgesehen, daß ein aktiver Kommunikationspartner nach Absetzen einer Dienstanforderung die betreffende Antwort an einer bestimmten Mailbox abholen kann.

Der hohe Funktionalitätsgrad (MACRO-Charakter) der Dienste DNC und MDE äußert sich einerseits in der Zusammenfassung von ganzen Abfolgen von Basiskommunikationsdiensten zu einem MACRO-Dienst.

Zum anderen wird der MACRO-Charakter in der hohen Anwendungsorientiertheit der DNC- und MDE-Dienste deutlich. Anwendungsorientiert heißt, daß neben der Information zur Abwicklung der Kommunikation

noch zusätzlich applikationsspezifische Information wie Arbeitsplatzparameter (Maschinennummer, Maschinengruppe) bzw. Auftragsparameter (Auftragsnummer, Arbeitsgangnummer, Splittnummer) mit übergeben werden können.

m-dnc

Mit der Funktion m_dnc werden einer Applikation DNC-Dienste mit sehr hohem Funktionalitätsgrad zur Verfügung gestellt. Dabei handelt es sich um Dienste, die typischerweise auf Daten anwendbar sind, wie beispielsweise

- Übertragen/Rückübertragen von Auftragsdaten, NC-Programmen, Werkzeugkorrekturdaten, RC-Programmen usw.,
- Starten von NC-Programmen, RC-Programmen usw.

Dabei ist die Funktionalität dem Anwender in einer ihm gewohnten Sprache zugänglich. Ein Prosatext wie "ich möchte eine DNC-Funktion ausführen, nämlich den Download des NC-Programms P124078 in die NC-Steuerung, die über den Kommunikationskanal Channel_1 ansprechbar ist" ist quasi 1 : 1 in einen Prozeduraufruf übersetzbar:
m_dnc (Channel_1, DOWNLOAD, NC_PROG, "P124078")
Funktion channel std_action type name

channel: logischer Kommunikationskanal, der mit m_open geöffnet worden war, und der nach m_connect mit einer entfernten Applikation verbunden ist.
std_action: Spezifiziert den auszuführenden DNC-Dienst (Standardgeräteoperation). Bei m-dnc existieren 3 Arten:
Synchrone Dienstanforderung (z. B. UPLOAD)
Asynchrone Dienstanforderung (z. B. UPLOAD_REQ)
Diensterbringung (z. B. UPLOAD_RSP)
type: Beschreibt die Art der Nutzinformation, die übertragen wurde oder die zu übertragen ist (Standarddatentypen).
name: Enthält den Namen der übertragenen oder der zu übertragenden Nutzinformationen. Hierbei kann es sich um projektierte Objekte handeln.

Die Standardgeräteoperationen sind die Dienste, die ein Fertigungsgerät von einem anderen anfordert. Fertigungsgeräte können Rechner (CC), Speicherprogrammierbare Steuerungen (SPS), Numerische Steuerungen (NC) und Robotersteuerungen (RC) sein. Die Dienste sind in der Regel rechnerinitiiert, einige wenige sind dagegen steuerungsinitiiert.

Datentransfer und Dateihandling

DOWNLOAD: Laden von Daten in ein entferntes Fertigungsgerät,
UPLOAD: Laden von Daten aus einem entfernten Fertigungsgerät,
REQUEST: Steurungsinitiiertes Laden von Daten,
BACKLOAD: Steuerungsinitiiertes Zurückladen von Daten,
DELETE: Löschen von Daten im entfernten Fertigungsgerät,
RELOAD: Nachladebetrieb bei Fertigungsgeräten mit zu kleinem Datenspeicher initiieren,
FILEGET: Datei von einem entfernten Fertigungsgerät auf das lokale Gerät kopieren,
FILEPUT: Datei vom lokalen Fertigungsgerät in das entfernte Gerät kopieren,
RENAME: Umbenennen von Dateinamen in entfernten Fertigungsgeräten,
DIR: Auflisten der in einem entfernten Fertigungsgerät gespeicherten Dateien.

Programmsteuerung

START: Starten eines Programms in einem entfernten Fertigungsgerät,
RESET: Beenden eines Programms in einem entfernten Fertigungsgerät,
BREAK: Unterbrechen eines Programms in einem entfernten Fertigungsgerät,
CONTINUE: Fortsetzen eines unterbrochenen Programms in einem entfernten Fertigungsgerät,
REPEAT: Wiederholen eines Programms in einem entfernten Fertigungsgerät,
SEQUENCE: Starten einer Programmfolge in einem entfernten Fertigungsgerät.
Für DNC sind u. a. folgende Standarddatentypen unterscheidbar:
SPS_PROG: SPS-Programm
NC_PROG: NC-Programm
U_PROG: Unterprogramm
TOOL_DATA: Werkzeugdaten
AUFTR_DATA: Auftragsdaten
MESS_PLAN: Meßplan
MESS_DATA: Meßergebnis
RC_PROG: RC-Programm

RC_DATA: RC-Daten
TEST_PROG: Prüfprogramm
APPL_SPEC: Applikationsspezifische Daten.

Die DNC-Dienste besitzen außerdem eine Schnittstelle zu Datenbanken, über die z. B. Abfragen in Tabellen nach NC-Programmen usw. ausgeführt werden können (u_db(ASK_DNC)).

m-mde

Mit der Funktion m_mde werden einer Applikation MDE-Dienste mit sehr hohem Funktionalitätsgrad zur Verfügung gestellt. Dabei handelt es sich um Dienste, die typischerweise auf Variablen, Zustände usw. anwendbar sind, wie beispielsweise:

- SPS-Eingänge/Ausgänge/Merker setzen bzw. lesen,
- Spindeldrehzahl, Vorschub auslesen,
- Maschinenzustände, Störungen melden,
- Stückzahlen/Ausschuß melden.

Dabei ist die Funktionalität dem Anwender in einer ihm gewohnten Sprache zugänglich. Ein Prosatext wie "ich möchte eine MDE-Funktion ausführen, nämlich das Überschreiben des Merkers 4711 in der SPS-Steuerung, die über den Kommunikationskanal Channel_2 ansprechbar ist, mit dem Wert 77" ist quasi 1 : 1 in einen Prozeduraufruf übersetzbar:
m_mde (Channel_2, WRITE, MERKER, "4711", 77)
Funktion channel std_action type name value.

channel: Logischer Kommunikationskanal, der mit m_open geöffnet worden war, und der nach m_connect mit einer entfernten Applikation verbunden ist,
std_action: Spezifiziert den auszuführenden MDE-Dienst (Standardgeräteoperation).
Bei m-mde existieren 3 Arten:
Synchrone Dienstanforderung (z. B. WRITE),
Asynchrone Dienstanforderung (z. B. WRITE_REQ),
Diensterbringung (z. B. WRITE-RSP),
type: Beschreibt die Art der Nutzinformation, die übertragen wurde oder die zu übertragen ist (Standarddatentypen),
name: Enthält den Namen der übertragenen oder der zu übertragenden Nutzinformation. Hierbei kann es sich um projektierte Objekte handeln,
value: Enthält den Wert der übertragenen oder der zu übertragenden Variablen/Nutzinformation.

Für MDE sind folgende Standardgeräteoperationen unterscheidbar:

Variablen- und Zustandshandling

STATUS: Abfrage von Zuständen eines entfernten Fertigungsgeräts,
READ: Variablen lesen (rechner- oder steuerungsinitiiert),
WRITE: Variablen schreiben (rechner- oder steuerungsinitiiert),
USTATUS: Melden von Änderungen von Zuständen bzw. Variablenwerten (steuerungsinitiiert),
IDENT: Identifizierungsinformation eines entfernten Fertigungsgeräts anfordern (Hersteller, Typbezeichnung).

Event- und Alarmhandling

ENABLE: Ereignis aktivieren,
DISABLE: Ereignis deaktivieren,
WATCH: Überwachung von Variablen oder Zuständen, wobei eingetretene Änderungen automatisch von dem entfernten Fertigungsgerät gemeldet werden,
ALARM: Absetzen eines Alarms (steuerungsinitiiert),
ACK: Quittieren eines Alarms (steuerungsinitiiert).

Bedienerdialog

INPUT: Der übergeordnete Rechner fordert die Eingabe von Textzeilen von einem Bedienerterminal an,
OUTPUT: Ausgabe von Textzeilen durch den übergeordneten Rechner auf ein Bedienerterminal,
GET: Ein Bedienerterminal fordert die Eingabe von Textzeilen beim übergeordneten Rechner an (Bedienerinitiative),
PUT: Über Bedienerterminal werden Textzeilen in den übergeordneten Rechner eingegeben.

Für MDE sind u. a. folgende Standarddatentypen unterscheidbar:
STÖRUNG: Störungsgrund
STÜCKZAHL: Stückzahl, Ausschuß
STATUS: Betriebszustand
MERKER: Merker
EINGANG: Eingang

AUSGANG: Ausgang
PROC_ZUSTAND: Prozeßzustand
MELDUNG: Meldung
TEMP: Temperatur
DRUCK: Druck
ZÄHLER: Zähler
VORSCHUB: Vorschub
SPINDEL_DZ: Spindeldrehzahl
APPL_SPEC: Applikationsspezifische Daten.

Die MDE-Dienste besitzen außerdem eine Schnittstelle zu Datenbanken, über die z. B. Einträge in Tabellen von Störungsmeldungen usw. (u_db(ADD_MDE)) gemacht werden können.

m-comm-serve

Mit dieser Funktion stellt der passive Kommunikationspartner die Bedienung einer gewünschten Zahl angekommener Messages sicher (Empfangene Indications, Confirmations). Sind keine Messages in der Empfangswarteschlange bzw. Mailbox, erhält die Applikation sofort die Kontrolle wieder zurück. Sind Messages in der Empfangswarteschlange, wird die von der Applikation angegebene Anzahl bedient. Es kann auch "Bedienen bis Empfangswarteschlange leer" gewählt werden. Zur Übergabe der Messages an die Applikation werden automatisch die betreffende Funktion u_dnc bzw. u_mde aufgerufen. Die Aktion, die darauf folgen soll, ist vom Anwender zu programmieren. Die Funktion m_comm_serve wird typischerweise zur Realisierung des aktiven Kommunikationspartners bei asynchronen Dienstanforderungen verwendet. Sie kann aber auch vom passiven Kommunikationspartner anstelle von m_server verwendet werden.

m-server

Mit dieser Funktion stellt der passive Kommunikationspartner die ständige Bedienung ankommender Messages sicher (Empfangene Indications, Confirmations).

Diese Funktion wird typischerweise zur Realisierung von Serverprozessen (Daemons) verwendet. Sie stellt eine Endlosschleife dar, innerhalb derer auf ankommende Messages gewartet wird. Bei Multitaskingbetriebssystemen wird der betreffende Serverprozeß schlafen gelegt, solange keine Messages ankommen. Zur Übergabe der Messages an die Applikation werden von EASYCOM automatisch die betreffende Funktion u_dnc bzw. u_mde aufgerufen. Die Aktion, die darauf folgen soll, ist vom Anwender zu programmieren. Die Funktion m_server kann über die Eingabe von Control_C wieder verlassen werden.

m-client

Mit dieser Funktion stellt die Applikation einerseits wie in der Funktion m_server die ständige Bedienung ankommender Messages sicher (Empfangene Indications, Confirmations), andererseits kann sie die vorbereitete einfache Schnittstelle zur Visualisierung benutzen. Diese Funktion wird typischerweise zur Realisierung von menügesteuerten Clientprozessen verwendet. Sie stellt eine Endlosschleife dar, innerhalb derer die Funktionen m_comm_serve, u_menu und u_switch zyklisch aufgerufen werden. In u_menu kann der Anwender sein gewünschtes Hauptmenü spezifizieren und in Übereinstimmung dazu in u_switch die Reaktion auf eine bestimmte Eingabe angeben. Mit m_comm_serve werden ankommende Messages bedient. Zur Übergabe der Messages an die Applikation werden von der Kommunikationsschnittstelle 1 automatisch die betreffenden Funktionen u_dnc bzw. u_mde aufgerufen. Die Aktion, die darauf folgen soll, ist vom Anwender zu programmieren. Die Funktion m_client kann über die Eingabe von Control_c bzw. von "x" wie exit wieder verlassen werden.

u-dnc

Zur Übergabe der über einen Kommunikationskanal empfangenen DNC-spezifischen Messages (Indications XXX_IND, Confirmations YYY_CNF) an die Applikation wird von der Kommunikationsschnittstelle 1 automatisch diese Funktion aufgerufen. Die Aktion, die darauf folgen soll, ist vom Anwender innerhalb von u_dnc zu programmieren.

Der Aufruf

u_dnc (Channel_1, DOWNLOAD_CNF, NC_PROG, "P124078")

Funktion channel std_action type name

bedeutet beispielsweise, es wurde das NC-Programm P124078 in die NC-Steuerung, die über den Kommunikationskanal Channel_1 ansprechbar ist, per DOWNLOAD erfolgreich übertragen.

Für die Parameter channel, std_action, type, name sowie für Standardgeräteoperationen und Standarddatentypen gilt das für m-dnc auf S. 14—16 Gesagte. std_action: Spezifiziert den auszuführenden DNC-Dienst. Bei u-dnc existieren zwei Arten:

angekommene Dienstanforderungen (z. B. UPLOAD_WH)

angezeigte Dienstanforderung (z. B. UPLOAD_CNF).

u-mde

Zur Übergabe der über einen Kommunikationskanal empfangenen MDE-spezifischen Messages (Indications

XXX_IND, Confirmations YYY_CNF) an die Applikation wird von der Kommunikationsschnittstelle **1** automatisch diese Funktion aufgerufen. Die Aktion, die darauf folgen soll, ist vom Anwender innerhalb von u_mde zu programmieren.

Der Aufruf

u_mde (Channel_2, WRITE_IND, MERKER, "4711", 77)

Funktion channel std_action type name value

bedeutet beispielsweise, der Merker 4711 soll in der SPS-Steuerung, die über den Kommunikationskanal Channel_2 ansprechbar ist, mit dem Wert 77 überschrieben werden.

Für die Parameter channel, type, name, value sowie für Standarddatentypen und Standardgeräteoperationen gilt das für m-mde auf S.6 und 8 Gesagte. Std_action: Spezifiziert den auszuführenden MDE-Dienst. Bei u-mde existieren zwei Arten:

angekommene Dienstanforderung (z. B. WRITE_IND)

angezeigte Diensterbringung (z. B. WRITE_CNF).

u-db

Diese Funktion ist Teil der vorbereiteten Schnittstelle zu Datenbanken. Sie wird für den Anwender aufgerufen, wenn innerhalb von DNC/MDE-Operationen Datenbankzugriffe erforderlich sind. Die Aktion, die darauf folgen soll, ist vom Anwender zu programmieren.

Es sind folgende Standarddatenbankoperationen unterscheidbar:

ASK_DNC: Abfragen von Datenbank-Tabellen nach DNC-spezifischer Information (z. B. NC-Programm),

ADD_DNC: Einträge von DNC-spezifischer Information in Datenbank-Tabellen,

ASK_MDE: Abfragen von Datenbank-Tabellen nach MDE-spezifischer Information,

ROD_MDE: Einträge von MDE-spezifischer Information in Datenbanktabellen (z. B. Störgründe),

ERROR: Ausgabe einer Datenbankspezifischen Fehlermeldung auf ein Bedienerterminal (z. B. "NC-Programm nicht freigegeben").

u-menu

Diese Funktion ist Teil der vorbereiteten Schnittstelle zur Visualisierung. Sie wird typischerweise zur Realisierung von menügesteuerten Clientprozessen verwendet. Innerhalb von m_client wird u_menu zyklisch für den Anwender aufgerufen. Der Code für das gewünschte Menü ist vom Anwender zu schreiben.

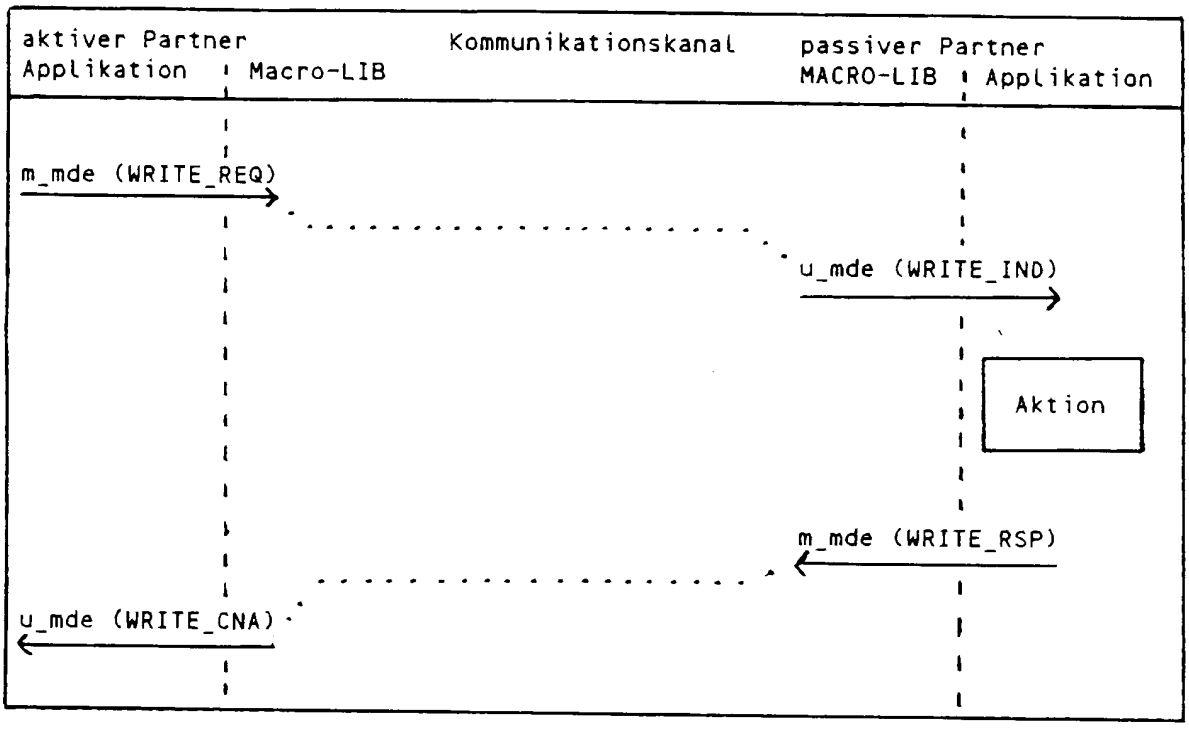
u-switch

Diese Funktion ist Teil der vorbereiteten Schnittstelle zur Visualisierung. Sie wird typischerweise zur Realisierung von menügesteuerten Clientprozessen verwendet.

Innerhalb von m_client wird u_switch zyklisch für den Anwender aufgerufen. Der Code ist vom Anwender in Einklang zu u_menu zu schreiben.

Die Anwendungsgebiete zerfallen in Bibliotheksaufrufe (m_xxxx Calls) einerseits und in Benutzerfunktionen (u_yyyy Calls) andererseits. Die Bibliotheksaufrufe werden von der Applikation aufgerufen, um sich den Dienst von der Schnittstelle erbringen zu lassen. Die Benutzerfunktionen werden von der Schnittstelle für die Applikation aufgerufen, codiert werden müssen sie aber vom Anwender.

Der Zusammenhang zwischen Bibliotheksaufgaben und Benutzerfunktionen ist nachstehend beispielhaft für den MDE-Dienst Merker schreiben (WRITE) gezeigt:



Die Applikation des aktiven Partners übergibt eine Anforderung Merker schreiben an die Schnittstelle **1** (MACRO-Library). Diese sorgt dafür, daß die Anforderung unter Durchlaufen des unterlagerten Kommunikationsprofils (vgl. **Fig.** 1, 3, 4, 5, 6, 7, 8, 9, 10) zum passiven Partner gelangt, wo die dortige Schnittstelle **1** sie an ihre Applikation weitergibt. Die Applikation führt dann die notwendige Aktion aus, die in einer vom Anwender zu kodierenden Prozedur programmiert ist, und teilt das entstehende Ergebnis wieder der Schnittstelle **1** mit. Diese sorgt ihrerseits dafür, daß die Ergebnismitteilung unter Durchlaufen des unterlagerten Kommunikationsprofils den aktiven Partner erreicht, wo sie über die Schnittstelle **1** zur initiierenden Applikation gelangt.

Die Schnittstelle **1** ist protokollunabhängig designed, so daß die Applikation beim Wechsel auf ein anderes Kommunikationsprotokoll nicht umgeschrieben werden braucht. Sie muß nur mit der passenden Bibliothek neu gebunden werden. Die Schnittstelle **1** ist weiterhin so designed, daß mit ihr viele unterschiedliche Kommunikationsaufgaben schnell und einfach gelöst werden können. Im folgenden sind einige Beispiele für deren Einsatz gezeigt.

Applikation aktiver Partner (asynchron)

m_start
m_open (CONNECT)
m_connect
m_dnc (UPLOAD_REQ)
Applikation läuft weiter, bis sie bereit ist, die Daten entgegenzunehmen
m_comm_serve
→ u_dnc (UPLOAD_CNF)
m_disconnect
m_close
m_finish

Applikation passiver Partner (Server)

m_start
m_open (LISTEN)
m_server
→ u_dnc (UPLOAD_IND)
Aktion
→ m_dnc (UPLOAD_RSP)
m_close
m_finish

Applikation aktiver Partner (synchron)

```
m_start
m_open (CONNECT)
m_connect
m_dnc (UPLOAD)
Applikation wartet
.
.
→ u_dnc (UPLOAD_CNF)
m_disconnect
m_close
m_finish
```

Applikation passiver Partner (2. Möglichkeit)

```
m_start
m_open (LISTEN)
m_comm_serve
→ u_dnc (UPLOAD_IND)
Aktion
→ m_dnc (UPLOAD_RSP)
m_comm_serve
m_close
m_finish
```

Applikation aktiver Partner (Client)

```
m_start
m_open (CONNECT)
m_connect
m_client
→ u_menu
→ u_switch
→ m_dnc (UPLOAD)
→ u_dnc (UPLOAD_CNF)
m_disconnect
m_close
m_finish
```

Patentansprüche

1. Kommunikationsschnittstelle für die offene Kommunikation von Anwendern mit Komponenten eines verteilten Kommunikationssystems, das in Schichten strukturiert ist, **dadurch gekennzeichnet**, daß die auf einer MACRO-LIBRARY basierende Kommunikationsschnittstelle (**1**) in Verbindung mit der obersten Schicht durch vier verschiedene Gruppen von Prozeduren bestimmt ist, von denen die erste Gruppe die von einem passiven oder aktiven Kommunikationsteilnehmer aufrufbaren Prozeduren zur Initialisierung oder Beendigung einer Kommunikation sowie zur Öffnung oder Schließung eines logischen Kommunikationskanals enthält, daß die zweite Gruppe nur von einem aktiven Kommunikationspartner aufrufbare Prozeduren mindestens zum Aufbau und Abbau einer Kommunikationsverbindung zu einem passiven Kommunikationspartner und zur Initiierung eines Datenaustausches enthält, daß die dritte Gruppe eine nur von einem passiven Kommunikationsteilnehmer aufrufbare Prozedur zur Bedienung ankommender Informationen und Beantwortung eines entfernt initiierten Datenaustauschs aufweist und daß die vierte Gruppe von einem Anwender zu kodierende Prozeduren, die von der MACRO-Library aufgerufen werden, aufweist und die Prozeduren zur Spezifizierung eines Hauptmenüs, einer Reaktion auf eine bestimmte Eingabe und zur Angabe auszuführender Funktionen enthält.

2. Kommunikationsschnittstelle nach Anspruch 1, dadurch gekennzeichnet, daß drei vom Anwender kodierbare und von der MACRO-Library aufgerufene Prozeduren vorgesehen sind, von denen eine auf Datenbankoperationen bezogen ist und von denen zwei auf Standarddatentypen und Standardgeräteoperationen bezogen sind und die die Reaktion auf eingegangene Dienstanforderungswünsche entfernter Kommunikationspartner beschreiben.

3. Kommunikationsschnittstelle nach Anspruch 1 oder 2, dadurch gekennzeichnet, daß eine Prozedur zur Initialisierung durch Einlesen von Projektierinformationen aus Konfigurationsdateien, zum Setzen von Kommunikationsparametern sowie zur Angabe der anzusprechenden Geräte und deren zugreifbare Objekte vorgesehen ist.

4. Kommunikationsschnittstelle nach einem oder mehreren der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß eine Prozedur vorgesehen ist, mit der die bestehenden Kommunikationsverbindungen unter Abmeldung aller angemeldeten Kommunikationskanäle und unter Rückgabe belegter Speicherplätze

in einem definierten Zustand zurückversetzbar sind.

5. Kommunikationsschnittstelle nach einem oder mehreren der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß eine Prozedur zur Öffnung eines logischen Kommunikationskanals unter Vergabe einer logischen Kanalnummer und zur Schaffung der Voraussetzungen für die weitere Kommunikation sowie eine Prozedur zur Schließung des logischen Kommunikationskanals unter Rückgabe der für die Kommunikation belegten Ressourcen vorgesehen sind.

6. Kommunikationsschnittstelle nach einem oder mehreren der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß je eine Prozedur zum Aufbau und Abbau einer Kommunikationsverbindung zu einer entfernten Partnerapplikation und eine Prozedur zum Abbrechen der Kommunikationsverbindung zu einer entfernten Partnerapplikation vorgesehen sind.

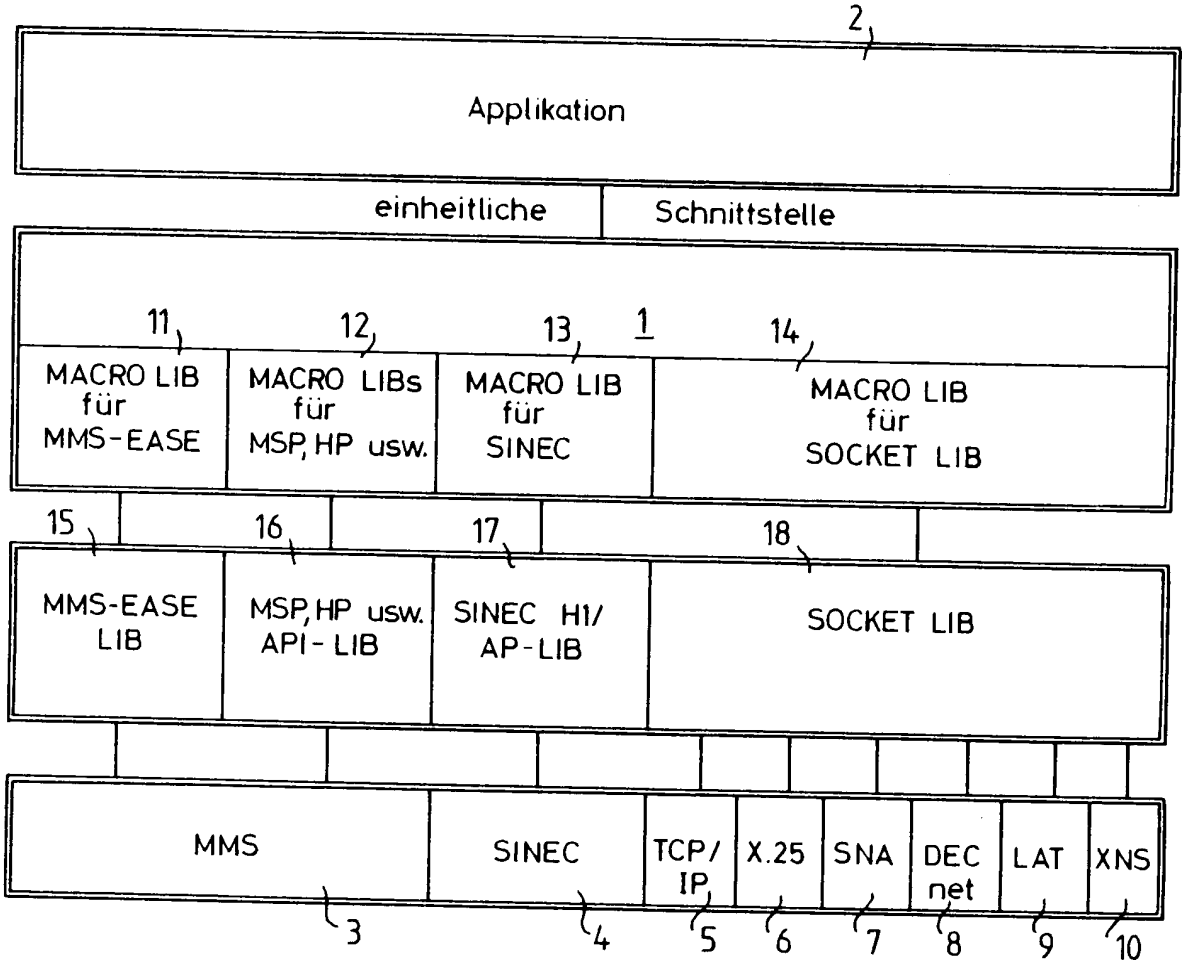
7. Kommunikationsschnittstelle nach einem oder mehreren der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß zwei Prozeduren in Form von Library Calls vorgesehen sind, die die Ausführung eines Dienstes von einem entfernten Kommunikationspartner anfordern unter Angabe eines logischen Kommunikationskanals, des auszuführenden Dienstes, der Art der zu übertragenden Nutzinformation und des Namens der Nutzinformation.

8. Kommunikationsschnittstelle nach einem oder mehreren der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß zwei Prozeduren für die Bedienung von ankommenden Kommunikationswünschen zur Verfügung stehen, wobei eine davon die Erstellung von Netzwerksreserveapplikationen in einfacher Weise erlaubt.

Hierzu 3 Seite(n) Zeichnungen

Nummer: DE 41 06 793 A1
Int. Cl.[5]: G 06 F 13/14
Offenlegungstag: 17. September 1992

FIG.1

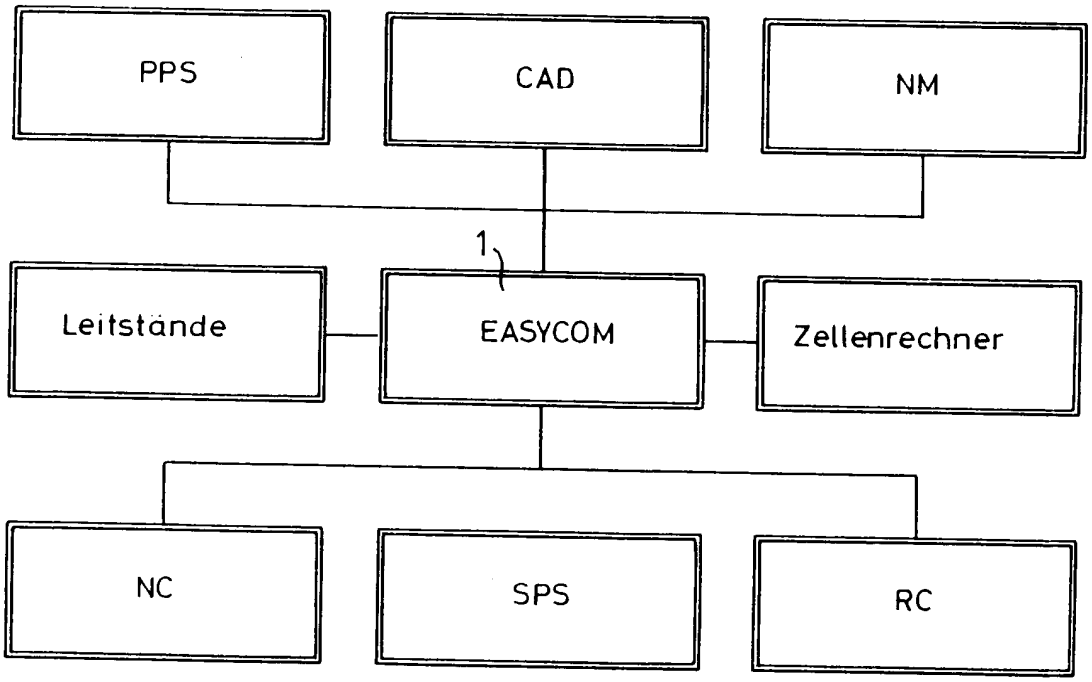


Protokollunabhängigkeit der MACRO-Kommunikations-schnittstelle (EASYCOM)

Die Reihe der Protokolle ist nach rechts fortsetzbar um weitere, etwa Bitbus, Profibus, FTAM usw.

Nummer: DE 41 06 793 A1
Int. Cl.[5]: G 06 F 13/14
Offenlegungstag: 17. September 1992

FIG.2



Berücksichtigte Anwendungsbereiche von EASYCOM

Nummer: DE 41 06 793 A1
Int. Cl.[5]: G 06 F 13/14
Offenlegungstag: 17. September 1992

FIG.3